Yann Kieffer
Grenoble-INP Esisar : $3^{ième}$ année IR&C

mardi 1^{er} juin 2021
14h00 → 15h30

MA351 : Théorie des graphes

Examen final

Calculatrices et tous documents autorisés

Les barèmes indiqués sont purement indicatifs.

1. Degrés des sommets dans les graphes non-orientés (4 points)

1) En utilisant l'encodage d'un graphe non-orienté sous forme de liste de voisinages, démontrez que :

$$\sum_{v \in V} d(v) = 2|E|$$

2) Existe-t-il des graphes non-orientés sur 5 sommets dont tous les sommets soient de degré égaux à 3 ?

2. Décompte des composantes connexes d'un graphe non-orienté (8 points)

Voici maintenant la routine A :

```
A :
Entrée: G = (V,E) avec E = {e_1,...,e_m}
  F ← ∅
  Pour i de 1 à m, Faire:
    si (V, F ∪ {e_i}) est sans cycle, faire:
      F ← F ∪ {e_i}
  Retourner F
 Fin
```

1) Illustrez le déroulement de cet algorithme sur un graphe ayant au moins 2 composantes connexes et 2 arêtes.

2) En utilisant le fait qu'à l'issue de l'algorithme, le graphe $H = (V, F)$ est sans cycle, démontrer que le nombre de composantes connexes de H (noté $p(H)$) vérifie la relation :

$$|V| = |F| + p(H)$$

3) Démontrez que toute composante connexe de H est incluse dans une composante connexe de G.

4) En déduire que les composantes connexes de H qui ont une intersection non-vide avec une composante connexe C de G forment une partition de C.

5) Montrez que si C, une composante connexe de G, contient plusieurs composantes connexes de H, alors il y a une arête e dans G qui n'est pas dans l'ensemble F retourné par l'algorithme.

6) Montrer que sous les hypothèses de la question précédente, l'arête e aurait due être choisie par l'algorithme.

7) Démontrez qu'à l'issue de l'algorithme, H a exactement les mêmes composantes connexes que G.

8) En déduire qu'il est possible d'utiliser cet algorithme pour compter les composantes connexes de G. Indiquez comment récupérer le nombre de composantes connexes de G !

3. Implémentation efficace de l'algorithme précédent (8 points)

Voici une implémentation efficace en Python de l'algorithme précédent :

```
def retrouve(f, x):
  if f[x] == x:
    return x
  else:
    return retrouve(f, f[x])

def identite(s):
  f = {}
  for x in s:
    f[x] = x
  return f

def composantes(g):
  f = identite(g)

  for x in g:
    for y in g[x]:
      a = retrouve(f, x)
      b = retrouve(f, y)
      if a != b:
        f[y] = x
  return f

def presente(f, g):
  c = {}
  for i in g:
    c[i] = []

  for i in g:
    j = retrouve(f, i)
    c[j].append(i)

  d = {}
  for i in c:
    if c[i] != []:
      d[i] = c[i]

  return d
```

L'objet de cette partie est de montrer que cet algorithme retourne bien les composantes connexes du graphe passé en entrée.

Voici un exemple d'exécution :

```
>>> g
{0: [2], 1: [], 2: [0], 3: [], 4: [], 5: [], 6: [], 7: [], 8: [9],
    9: [8, 11], 10: [], 11: [9]}
>>> z = composantes(g)
>>> z
{0: 0, 1: 1, 2: 0, 3: 3, 4: 4, 5: 5, 6: 6, 7: 7, 8: 8, 9: 8, 10: 10, 11: 9}
>>> c = presente(z, g)
>>> c
{0: [0, 2], 1: [1], 3: [3], 4: [4], 5: [5], 6: [6], 7: [7], 8: [8, 9, 11], 10: [10]}
>>>
```

1) Si g est un dictionnaire sur un ensemble de clés S, et x est une clé de S, que retourne `retrouve(identite(`g`),` x`)` ?

2) Si on considère la fonction partielle f, définie sur l'ensemble V, comme l'encodage d'un graphe orienté

(V, A) avec $A = \{(v, f(v)), v \in V\}$, que calcule exactement la fonction $\texttt{retrouve}(f, v)$?

3) Déroulez l'appel de la fonction `composantes()` sur un graphe de votre choix ayant 2 arêtes et 4 sommets.

Dans la suite, notre objectif sera de montrer qu'à l'issue de l'algorithme `composantes()`, deux sommets u et v de G seront dans la même composante connexe, si et seulement si $retrouve(f, u) = retrouve(f, v)$, où f dénote la fonction partielle retournée par `composantes()`.
Notre méthode sera de montrer que cette propriété est vérifiée à chaque étape de l'algorithme pour un graphe bien choisi.

4) Montrer qu'après l'appel $f = \text{identite}(g)$, deux sommets u et v du graphe $(V, \emptyset)$ sont dans la même composante connexe si et seulement si $f[u] == f[v]$.

5) Montrer qu'après les 4 instructions Python :

```
a = retrouve(f, x)
b = retrouve(f, y)
if a != b:
  f[y] = x
```

les sommets x et y retourneront la même valeur pour la fonction $u \to \texttt{retrouve}(f, u)$.

6) En déduire qu'il existe un ordre sur les arêtes de G (à préciser !) $E = \{e_1, \ldots, e_m\}$ tel qu'après k passages dans la boucle de la fonction `composantes`(), les sommets u et v sont dans la même composante connexe du graphe $G_k = (V, E_k)$ avec $E_k = \{e_1, \ldots, e_k\}$ si et seulement si $\texttt{retouve}(f, u) == \texttt{retrouve}(f, v)$.

7) En déduire que pour la fonction f retournée par `composantes()`, u et v sont dans la même composante connexe de G si et seulement si $\text{retrouve}(f, u) = \text{retrouve}(f, v)$.

8) Expliquez ce que retourne la fonction `presente(f, g)`, lorsqu'on lui passe en premier argument le résultat f de l'appel `composantes(g)`.

9) Détaillez le fonctionnement du code de la fonction `presente()`.

Exc 1 :

1) On veut $\sum_{v \in V} d(v) = 2|E|$

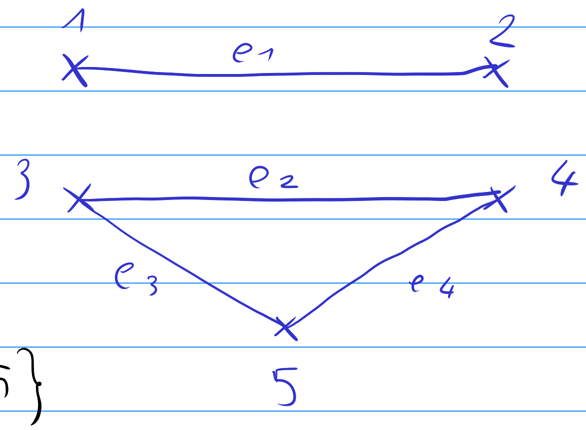
Correction : Chaque arrête $\{x, y\}$ de E augmente de 1 la valeur de $d(x)$ et de 1 la valeur de $d(y)$. Donc de 2 la valeur de $\sum_{v \in V} d(v)$.

2) $5 \times 3 = 15 = \sum_{v \in V} d(v)$ est impair donc ne peut pas être égal à $2|E|$.

Donc $\boxed{\text{non}}$

Exc 2 :

1)



$V = \{1, 2, 3, 4, 5\}$

$F = \{e_1, e_2, e_3, e_4\}$

$F \leftarrow \emptyset$

$i \leftarrow 1$

$(V, \{e_1\})$ est sans cycle

$F \leftarrow \{e_1\}$

$i \leftarrow 2$

$(V, \{e_1, e_2\})$ est sans cycle

$F \leftarrow \{e_1, e_2\}$

$i \leftarrow 3$

$(V, \{e_1, e_2, e_3\})$ est sans cycle

$F \leftarrow \{e_1, e_2, e_3\}$

$i \leftarrow 4$

$(V, \{e_1, e_2, e_3, e_4\})$ est <u>avec</u> cycle

On renvoi F, $\{e_1, e_2, e_3\}$

2) On veut montrer que : Comme $H=(V,F)$ est sans cycle, on a :

$$|V| = |F| + p(H) \rightarrow \text{nb de composantes connexes de } H.$$

Preuve : 2 idée :

① Pour chaque composante connexe sans cycle X,
$|X| = |F'| + 1$ ↳ nb d'arrête de la composante connexe

donc $|V| = \underbrace{\Sigma |F'|}_{|F|} + \underbrace{1}_{p(H)}$

② Par récurrence (+ rigoureuse)

$n = |F|$, le nb d'arrête d'un graphe sans cycle

Initialisation :

Soit $H=(V,\emptyset)$ un graphe sans arrête
alors $p(H) = |V|$, on a bien $|V| = |\emptyset| + |V|$

Hérédité :

Soit $H=(V,F)$ un graphe sans cycle tq :
$|V| = |F| + p(H)$

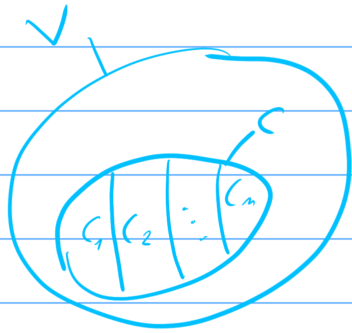
Soit $\{x,y\}$ une arête hors de F, et telle que
$H' = (V, F \cup \{x,y\})$ soit sans cycle.
Alors H' étant sans cycle, $\{x,y\}$ a rassemblé 2 composantes connexes en 1 seule (sinon H' aurait un cycle).
$p(H') = p(H) - 1$ donc $|V| = |F \cup \{x,y\}| + p(H')$

3) Soit C une composante connexe de H.
Soit $x \in C$, alors $x \in C'$ où C' est une composante connexe de G

H est G sans cycle

Il faut montrer $C \subset C'$

Soit $y \in C$ ⚠ à réfléchir.



4) Soit C une composante connexe dans G et $C_1, C_2, \ldots, C_n$ les composantes connexes de H qui ont une inter non vide avec C.

On veut montrer : 1) $\bigcup_{i=1}^{n} C_i = C$

2) $\forall i,j \in [\![1,n]\!], i \neq j \Rightarrow C_i \cap C_j = \emptyset$

1)